4-7-2000

It's grammatical!

Kurt Stoskopf

Rochester Institute of Technology

A Thesis submitted to the Faculty of the College of Imaging Arts and Sciences
in candidacy for the degree of Master of Fine Arts.

It's Grammatical!

By
Kurt Stoskopf

April 7, 2000

APPROVALS

Chief Advisor

Professor Jim Ver Hague

Date: 7/5/00

Associate Advisor

Associate Professor Deborah Beardslee

Date: 5 July 2000

Associate Advisor

Professor Bob Keough

Date: 7-5-00

Chairperson, School of Design

Professor Nancy Ciolek

Date: 7.12.00

I, Kurt Stoskopf, prefer to be contacted each time a request for production is made.
I can be reached at the following address:

Date: 7-18-00

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I wish to thank my parents, Lawrence and JoAnn Stoskopf, for being there for me every step of the way. I could not have done it without their support. In addition, thanks go to my thesis committee, Jim Ver Hague, Bob Keough, Deborah Beardslee, and Jack Slutzky, as well as to the faculty at the Applied Art & Computer Graphics Department. Thank you for your patience and understanding as I wandered down the long path towards accomplishing my goals. I also want to thank all my friends for being an important part of my life. You know who you are.

Chapter 1

INTRODUCTION

"What the heck is EDUTAINMENT?!"

The summer before I began working on my thesis, I was working as an intern in New York City for the investment and banking company, J.P. Morgan. While relaxing after a hard day at work, I was discussing possible ideas for thesis projects with my roommate, who was also a graduate student in Deaf Studies at Columbia University. He had been trying to come up with ideas for a topic related to teaching English to deaf students. He suggested a joint project in which we would develop an English educational program for Junior High school students. The goal would be to create a program that would actually encourage students to learn English, rather than playing games while being taught, as they were doing in the edutainment programs that were currently available for classroom use. When I heard that term, I was curious because I had never heard the term edutainment before.

My roommate explained that edutainment was a term used to describe software that was a combination of education and gaming. He said that many teachers he had worked with viewed edutainment as being bad for the students because it put the focus on game-play rather than on the learning process itself. Teachers thought that edutainment tried to replace the teacher, rather than trying to work with the teacher. After agreeing about the benefits of an educational program that was not edutainment with my roommate, we decided that we were going to use his idea for a joint thesis project. We would create a program that taught junior high school students by using a more subject-oriented focus. We planned to accomplish this by creating a program that avoided the use of edutainment. Our goal was to create a program that is used as a tool by teachers as part of their in-class teaching strategies.

Statement of thesis problem

There is a lack of software on the market that actively addresses the issues of learning and applying English skills in an environment that encourages learning and English language development instead of focusing on game play. Much of the available software, such as *Reader Rabbit 2000* and *Reading Blaster*, that carries the label of "educational software" is really edutainment and consists of animated cartoons, games, or puzzles that are disguised as educational programming. Many of the English programs, especially reading software, read aloud the stories, sentences, or words studied to the students rather than requiring the students to read the material. My partner and I have identified a need for reading programs that focuses on reading skills rather than listening skills and can be used by teachers as part of the normal classroom curriculum in the middle school and junior high school levels. There is also a need for programs that focus on the reading and error identification approaches, instead of providing multiple choice questions and fill-in-the-blank approaches. In our thesis project, we will create an educational program that addresses these needs.

Significance of thesis problem

The prevalence of edutainment software is rapidly becoming a serious problem. It is altering the students' perception of learning, changing from that of having to work to learn and understand the subject, to depending on a program to explain and break down the subject matter into easily understood chunks of information. It is also altering peoples' perceptions of the value of computers in education. There is a growing belief that computers will soon replace teachers in the classroom, that computers would do a better job of presenting materials to students than a teacher could ever possibly do. When one uses edutainment software in the classroom, one runs a risk of "dumbing down" the educational system. The students will grow to expect that the computers will do everything for them, and that, in turn, would discourage the students from doing research and exploring the information at their own pace. The biggest danger edutainment software presents to the classroom environment is that it excludes the teacher from the learning process, and at the same time, precludes student/teacher interaction. The software program we developed addresses the need of a software program becoming a part of the classroom instruction strategy, rather than becoming something that exists independently of the classroom.

Constraints

There are many limitations inherent in developing an educational program. We have identified four major limitations that are significant to our thesis project. First, software programs eliminate the necessary human to human contact in the classroom; they can never truly replace teachers in the classroom. Second, software programs are by nature not intuitive. They are unable to deal with unexpected responses or input that they are not specifically programmed to handle. Third, we have a limited amount of time to create the program. We will have to create a prototype software program rather than a final, finished software program. Finally, software programs are not aware of external factors; they cannot get the true measure of a student's potential. It cannot identify underlying problems that could be preventing students from fully achieving their goals. The computer is merely a tool that people use to accomplish an end. It is not something that can set goals and develop strategies towards accomplishing those goals. Educational programs can teach, but cannot become the teacher.

Assumptions

For our thesis project, we are working under several assumptions that will determine its success or failure. We are developing this project with the understanding that the classrooms using the program will have at least one computer available for use. If there is no computer available, our project becomes irrelevant. The second assumption is that the program accompanies the materials that the teacher teaches in the classroom. Without support from the teacher, our project has very little educational value. We will be creating a prototype version of the program for the thesis project, with the goal of continuing this beyond the requirements for the thesis project. The program, as we designed it for the thesis project, will have three content areas covered on its focus on English education. The three content areas, Spelling, Tense, and Syntax may be too narrow to make this program useful for the entire school year. Finally, we will be focusing on error identification and requiring text-entry corrections to correct any identified errors. This method of input may preclude students with serious reading disorders and those with physical handicaps from successfully using the program.

Chapter 2

REVIEW OF RELATED LITERATURE

In the education systems used in the United States, there are always questions about how teachers and schools can provide better education and better learning environments for the students. When faced with growing costs of education and dwindling budgets, the administration starts looking for ways to cut costs, while at the same time, providing students with better work environments that subscribe to current-day educational buzzwords. In the last few years, the school systems have turned to computers as the solution to their problems.

How could a computer, a machine, be considered a viable alternative to teachers in the current educational system? Frank Smith writes in his book, *Insult to Intelligence: A Bureaucratic Invasion of Our Classrooms*, " . . . computers are the ideal devices for programmatic instruction. They are more attractive to children, more effective drill masters than teachers, and more easily controlled by administrators. They are even cheaper than teachers, whose salaries are the largest and least tractable component in the spiraling costs of education. The world has entered the age of computers, and schools will not be able to resist them." (Smith, 13). Computers are impersonal things; they do the things they are programmed to do efficiently. There are no personal biases or personal issues that would affect its behavior towards students. They do what they are supposed to do, nothing more.

These qualities are very attractive to teachers, administrators, and parents. Frank Smith refers to computer-based instructional media as "r-bbit" programs. These programs are "attractive to teachers because it keeps children quiet and occupied." (Smith, 8). They know exactly what they can expect from computer based educational programs. The computers run the same activities over and over again. Computers are attractive to teachers and parents because they provide an environment where "little direct supervision is required.' As an additional inducement to teachers who want to hand over all their responsibilities to the [software] publishers." (Smith, 7). Administrators also find the computer-based education appealing because the programs " . . . promise so much, take up so much time, and generate so many scores and comparisons." (Smith, 7). These scores and comparisons are used by the administration and teachers to determine if their students meet the minimum educational standards required by the state. They also can use the scores to justify their educational methods.

Given that computers appeal to the administrators, teachers, and parents' baser instincts, how is it that they appeal to students too? Frank Smith answers this question by stating, "It is not difficult to see why computer-based instructional programs are attractive to many children . . .. Children Prefer the r-bbit because schoolwork is presented like the Saturday morning television cartoon show. If children must engage in totally meaningless activities like filling in missing letters of random words, that is the way to do it. " (Smith, 8).

How does one identify a program that could be described as a "r-bbit?" Frank Smith describes an experience with educational software that led him to label educational software as the "r-bbit."

He was attending an educational convention, where he came across a demonstration of an educational program (Smith, 2).

> The teachers were, I found, observing a demonstration of a computerized reading lesson. They were gazing at a cartoon representation of a desert scene on the screen of a small desktop computer. There was a bright blue sky above, a yellow landscape below, a few cacti, and in the center of it all a rabbit with large floppy years (sic) and a mischievous look. In large print on the side of the screen were the letters r-bbit. A flashing message at the bottom of the screen was appealing: "Please tell me your name." (And if the appropriate accessories were purchased, the computer will speak its remarks aloud.)
>
> The demonstrator was trying to persuade someone to oblige this friendly computer. Eventually a teacher typed "Cynthia" on the keyboard, and the computer immediately responded, "Can you fill in the missing letter in
> r-bbit, Cynthia?"
>
> The demonstrator urged the teacher to give the wrong answer. (Promoters of educational programs are always anxious to demonstrate the remarkable things their systems do when mistakes are made.) "Type K," said the demonstrator helpfully. Cynthia typed K.
>
> Bells rang, lights flashed, the rabbit's ears drooped, and the computer announced, "Too bad, Cynthia, K is not the right answer. Would you like to try again?"
>
> "Type the right answer this time," suggested the demonstrator. "Type A." The teacher typed A.
>
> Once again bells rang and lights flashed, but now the rabbit's ears perked up and it began munching on a carrot that magically appeared from off-screen.
>
> "Well-done, Cynthia, " said the computer. "A is the correct answer. Your score so far today is 50 percent. Your score last time was 0 percent. That is a 50 percent improvement. Congratulations. Would you like to try another one? "

When Frank Smith refers to the "r-bbit," he is referring to the current trend of computer based instruction. He states, "I have chosen the r-bbit as the symbol of the programmatic approach to instruction which believes that children learn by practicing one systematic thing after another--"Fill in the blank and find out whether you're right or wrong." (Smith, 3).

Many software publishers are creating programs that try to make education fun by incorporating games, cartoons, and sounds in an educational environment. This growing trend in educational design has been labeled, "edutainment." Edutainment is defined as "A general classification for software programs that combine elements of instruction and entertainment, including video, animation, and music. Educators disagree on the educational value of most 'edutainment' software." (Edweek, Online).

Chapter 3

PROCEDURE

When we got the topic of the thesis approved and obtained all the necessary departmental approvals, we were all set to work on the thesis project. My partner and I talked on the telephone and the first thing we both said was, "What now?"

We started discussing the concept of the thesis project without committing ourselves to any one approach, except that the thesis was an English educational program geared towards junior high school students. We both talked about our desire to avoid the edutainment approach and making the program function as an integral part of the junior high school teachers' curriculum. We agreed to divide the responsibility of the thesis into two parts. My partner, since he was a graduate student at Columbia University and was working towards his master's degree in education, would handle the English skill problems and the text content. I would develop the actual interactive program and design the interface and layout of the program.

Once we got the basic responsibilities worked out, we boiled down our thesis objectives into distinct, prioritized goals and used those goals to guide the further development of the project. The goals as we established them were as follows:

- Program functions as an educational program that works with the teacher, not independent of the teacher.
- Program will focus on the material to be learned and practiced, not on the entertainment aspect. This program is not edutainment.
- Program focus on students' error identification and problem solving skills, not "click and play" approaches to education.
- Program will appeal to the largest number of students, not limiting itself to one specific fad.
- Functionality is the key. The program will be easy to use and will encourage students to focus on the work they are supposed to do.
- Program will work on a wide range of computers. It will be compact and cross-platform compatible, as well as working on the Internet.

Once these objectives were established, we started gathering research on educational software and experts' thoughts on the computer's use in education. When we were doing the research, my partner pointed out that one writer, Frank Smith, made some very valid points about the use of computers and edutainment in school, and how they were changing the way we think about

classroom education. His writings, especially his thoughts about the "R*bbit," provided us with the theoretical framework on which to proceed on designing the software. For further analysis of Frank Smith's philosophy and how it affected our designs, please refer to Chapter 2, Review of Literature. In short, our program would not provide students with pre-set questions with multiple-choice answers or fill in the blank sentences, such as, "The red fox _____ over the lazy brown dog." We would focus on the identification and correction of errors in a work environment that would simulate the actual writing and proofreading experience. In the case of English education, we would use a paragraph as the work environment.

The next step was to design a survey to get opinions from students on their interests and things that they wanted to have in the ideal educational program. With this survey, we were trying to identify things that would appeal to the students. When we identified things the students wanted to see in a software program, we would incorporate those things into our software program. This way, the program would be designed for the kids with suggestions from the kids, rather than having something forced on them. Please refer to Appendix A for a sample of the survey questions we asked students, and a summary of their responses. The results were rather interesting, to say the least. Many of the responses led us to believe that the ideal educational program would be an arcade game, where they would get the chance to kill, maim, or get a prize after every correct answer. Those responses showed us the damage edutainment has done to students. It has encouraged them to play games, rather than focusing on learning.

When my partner and I took a closer look at the survey results, questions 4, 7, and 8 provided us with the most useful information:

Question 4: What they consider cool

Question 4 focused on the things the students considered "cool" and gave us a starting point to identify elements that would attract them to the program. Most of the things they considered "cool" were articles of clothing, from particular brand names. Most of the clothes they selected, Polo, Boss, Nike, Gap, and Tommy Hilfiger showed us that the students liked things with strong crisp colors and clean typographical approaches to labels and words. Please see Appendix B for examples of Polo, Nike, Gap, Boss, and Tommy Hilfiger items.

Question 7: Areas that they want to get help on

Question 7 focused on the areas the students felt that they needed help. The results surprised us because they were very similar to what we had previously identified as essential parts of the thesis program. The four things they requested help with the most were: Creativity and Style with 9 votes, Editing and Proofreading with 7 votes, Grammar and Spelling with 6 votes apiece. We had to eliminate the Creativity and Style answers because that was beyond what the thesis project could teach, because computers can only respond based on what they are programmed to do. Creativity and Style has the potential for multiple permutations on wording and writing approaches, that it would be virtually impossible for us to tackle within the timeframe for the thesis project. We established the program as previously stated in our third goal for the thesis software program, using the Editing and Proofreading approach for the whole program. We used the Spelling and Grammar selections for the sections of the program. Grammar turned out to be a rather broad subject area, so we broke that down further into two sections, Grammar and Syntax, for the sake of clarity.

Question 8: Ideas and advice for us

Question 8 asked students for ideas and advice on what we should do in the program. We received a long list of responses. For a list of responses to this question, please refer to Appendix A. We identified the use of colors, especially bright and clear color usage as an important aspect of the program. They also demonstrated their desire for a fast program. 3-D images and the use of American Sign Language figured prominently in their responses, but we eliminated that because of size constraints. With this great set of responses and ideas, my partner and I agreed that we had a strong foundation to begin working on the actual thesis program.

Design

Before we started programming, I suggested that we do a storyboard of the program and get student responses from that before making the big effort to create all the graphics and programming the final thesis project. The storyboard would allow us to create a visual representation of the program on paper and allow us to make corrections and revisions on paper before committing to the time required to create the computer generated imagery. We could also reproduce the storyboards and have students write feedback on the storyboards, if they couldn't describe what they wanted to see in the program verbally. My partner agreed with me on the plan of approach and said that he needed to make a preliminary presentation to his chairperson in the next two weeks.

I went to New York City two weeks later to present the storyboards of the thesis project to a junior high school class that my partner was teaching, and to give a short presentation to my partner's chairperson. Please see Appendix C for samples of the first design and storyboards of the thesis project. I made some preliminary designs and printed them out for the presentations.

I designed the programs with a simple interface approach. I analyzed several programs available on the market and came to the conclusion that grouping the user controls to one side, while grouping the content and materials to be read opposite the control groupings appeared to be the most successful in terms of functionality and aesthetic designs. Considering the order of the groupings, I had a choice of using a vertical bar or a horizontal bar design for the interface. Looking at the four grouping possibilities,with the black bar representing the navigation interface, (figures 1, 2, 3, 4), I eliminated figure 2 and figure 3 due to monitor constraints. If the students used computers with small screens, or if they were viewing the program on a web browser, I ran the risk of having the user interface bar being cut off by the screen or browser window.

Figure 1
Screen with navigation bar on left

Figure 2
Screen with navigation bar on right

Figure 3
Screen with navigation bar at top

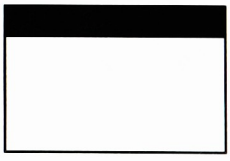
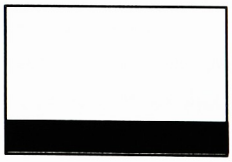


Figure 4
Screen with navigation bar at bottom



After narrowing the choices down to either figure 1 or figure 4, I quickly eliminated figure 4. According to basic design principles, the user control grouping at the top of the screen gave the screen design the feeling of being top-heavy and appeared to give the viewer a rather oppressive work environment. It also gave me the visual effect of limiting the usable area of the screen. Figure 1 gave me more area to work with, and the vertical user control bar gave me more screen real estate to work with. The text buttons could be added and not feel so crowded because the English text was read from left to right, not from top to bottom in the Asian countries. I could put more controls in the vertical user control bar than I could in the horizontal one.

After receiving via e-mail the samples of English problems my partner created, I had to figure out how I could show the paragraph while providing a space where students could enter their corrections for the errors they identified. I decided on using a pop-up dialog box to give the students more area to work on, while enhancing the focus on correcting the identified errors. Please refer to Appendix C for an example of the pop-up dialog box.

I also tried to develop a metaphor that would tie the project together, and decided on using a cartoon character that would give the students comments on the answers they gave. I created the storyboards for the software program and left for New York City.

Presentation

I arrived in New York City at 5 a.m. and met my partner before having to do the presentation at 8 a.m. for the students in Mrs. Hartman's classroom at St. Francis de Sales School for the Deaf in Brooklyn, New York. While walking to the school, we both went over the storyboards and my partner thought it was looking good. When we presented the storyboards to the students in the classroom, the teacher, principal, and superintendent were there to see our presentation. They were interested in new ideas for educational software, especially those that would be of benefit to their students.

We presented our idea for the educational software program we were developing for English skills, along with handing out copies of the storyboards to the students in the classroom and asked for their responses. The consensus was that they liked it! They commented that they liked the colors we used on the program as well as the layout scheme we had developed. They wanted to volunteer to test the program when it was completed. The faculty in attendance were impressed with the approach we took with the program and said that they were interested in using the program for their classes. After the presentation, we asked both students and faculty for their opinions on what they would like see improved for the final version of the thesis program. Their responses were that they wanted the program simplified some more, use less screens and pop-up dialogue boxes because they felt all of those were too distracting and had the potential to confuse the students. Some students also hated the cartoon and wanted the program done without the cartoon. We wrote down the responses we got from the students and the faculty.

With the presentation to the students completed, we went to give the presentation to my partner's chairperson at Columbia University. When we presented the thesis project and the storyboards to Dr. Kretschmer, my partner spoke to him for ten minutes, and I gave a short, ten-minute walkthrough of the thesis project. After we finished the presentation, he asked me for help with one of his programs. He was asking how he could design the interfaces for a program so that the user could understand and respond quickly to the questions being asked on the program.

I showed him a few possible design solutions, and he was happy with my suggestions. After we finished talking some more about the thesis project, he told my partner that he was finished with his masters' thesis, and would sign the approval papers for the completion of his thesis project, based on what was covered in our presentation. I was surprised at that, because I had yet to start the actual programming for the interactive program, and my partner was already done!

After finishing both presentations and on the train ride back to Rochester, I looked at the paper walkthrough and realized that we had gone through the entire process, talking about the program not being edutainment, but having a cartoon character prance about in the program, giving students congratulatory statements or incorrect statements, that this program was indeed edutainment. After a quick telephone call to my partner, we decided that, for my part, I would need to revise the program to eliminate the cartoon character and incorporate the comments we received from the presentations into the program. He wanted me to simplify the program. I agreed, after all, this was a design problem that would provide me with a good challenge. When I arrived home, there was a message on my machine from a local company offering me a full-time job doing web page design and creating illustrations for a series of clip art files for their line of nutritional software for schools. I had been working for the company as a freelance artist for several months preceding the job offer. They had been impressed with my skills and wanted me to work for them full time. I accepted the job and began working for the company the next day. I accepted the job without thinking about the consequences. I became a full time worker as well as a full time student. With the new job responsibilities, the time I had available for working on the thesis was reduced.

Development

Working at my job for a year and a half, and finding out about how people perceived the credibility of companies in the industry and the qualifications of the individual employees of the company, I realized that finishing graduate school was important if I wanted to help my company be more competitive in the industry. After this realization, I reorganized my priorities and started working on the thesis again.

My first step in developing the actual interactive program was to experiment with different authoring programs. I wanted to use an authoring program to develop the thesis project because the authoring programs provide an environment where I can create software programs rapidly by using pre-set elements that create code automatically and with minimal programming to add advanced features to the software program.. If I were to try creating my thesis without an authoring program, I would have to learn a programming language, such as C++ and write many lines of code to create the thesis program. With this in mind, I selected three leading software authoring programs: Macromedia Authorware, Macromedia Director, and Macromedia Flash. While using these programs, I quickly discovered that there were pros and cons for each of the authoring programs. Many of the cons were enough for me to stop using the programs altogether. To begin selecting the programs I would work with, I chose two of the most widely used multimedia authoring programs to begin my search. I chose Macromedia Director 7 and Macromedia Authorware 4.

I began working with Macromedia Authorware 4 because its advertisements claimed that the program was good for creating educational software programs. The object oriented programming system of Authorware was supposed to make the programming and development aspect of creating an interactive program easier by allowing the developer to create programs by combining icons that represented sections of code to create desired results. The icons represented sections of code,

and I could change the characteristics of each icon independently of other icons, hence the label "object oriented programming." I started playing around with the program and found that its timeline approach to designing software and its visual layout system was excellent for edutainment programs and ideal for keypress and multiple choice interaction. The timeline allowed me to create a hierarchical map of the software program, and allowed me to organize the program in sections. My only concern about the program was its lack of support for many kinds of graphics file formats I would be using to keep the size of graphics small, as well as the type of self-running programs it created. The projector files Authorware 4 created were rather large (over 800 KB) and very platform specific. I could only create projector files for the Macintosh computers because I was using a Macintosh computer to create the program. To create a projector file that would work on IBM PC compatible computers, I would have had to find a PC computer with a PC version of Authorware 4. Using this program to meet the objectives we set for the thesis project, I would have had to provide two separate files: one for the Macintosh and one for the IBM PC computers. The web support available for Authorware 4 was available widely only for the IBM PC computers and would not work on the Macintosh computers. After looking at the pros and cons of this program, I decided that this program was not the one that would allow me to achieve the objectives of the thesis project.

Looking at Macromedia Director 6 next, I found that it was slightly better for the objectives. Director 6 uses a totally different authoring approach than Authorware 4. Director 6 uses a movie-style approach to creating interactivity. Instead of using object oriented programming, Director 6 uses a programming language called Lingo to create interactions. Before investing time in experimenting with Director 6 and Lingo programming, I checked its authoring capabilities, and it had the same requirements as Authorware 4. To play back files, it required the developer to have the Director 6 program for the platform it was played on. This meant, in order for my programs to work on an IBM PC computer, I would have to have the IBM PC version of Director 6 too. This was something I wanted to avoid because I did not want to spend a lot of time and money to create two separate program files for the different computer systems. Director also created large program files. The player file that allows the user to run the software program without the Director program was over 1 MB in size , without the actual software program. This large file size was the deciding factor in rejecting this program. I wanted to keep my program file sizes as small as possible, so it could be used on a wide variety of mediums, such as the Internet, Floppy Disks, and on a range of computers.

A solution to this dilemma came to me from a different source. I had been learning a web graphics and animation program named Macromedia Flash 4 because more and more web designers and clients were looking for people with Flash animation skills. While experimenting with Flash 4, I discovered that it had programming capabilities, and that it could create self-running interactive programs. The wonderful thing about Flash 4 was that it could create platform independent player files. The only requirement was that the host computer has the player file on the computer. Most Web browsers that were newer than version 4.0 of Netscape Navigator and Microsoft Internet Explorer came with the Flash player and Flash plug-in pre-installed. From these discoveries, I decided that Flash 4 was the program of choice for the thesis project.

Macromedia Flash 4 is a vector animation program that is known for its fast playback and resolution independent graphics. The graphics Flash created can be scaled or reduced in size with no discernible loss of quality. The native vector graphic capability of the program proved to be the biggest advantage it had over Authorware 4 and Director 6. Flash could create its own graphics

and author interactive programs using its native graphics. Using the native graphics capability helped me modify the design of my paper walkthrough to the final interactive program quickly.

During this time, I received a better job offer and accepted it. I was now working as a visiting instructor for the National Technical Institute for the Deaf's Applied Art and Computer Graphics department. While working for the Applied Art and Computer Graphics department, I had more free time to work on developing the thesis program.

Redesign

With newfound motivation, I started working on the thesis project again with a fresher perspective on things. After I reviewed the designs from the paper walkthrough, shown in Appendix B, and incorporating the changes based on the comments from the students and faculty of the St. Francis de Sales School for the Deaf in Brooklyn, New York. I tried to develop my designs so that it would not be edutainment and make it simpler and more education-oriented. I decided that of the original design, I would keep the vertical user control grouping bar and the location of the visual elements on the page. I eliminated the cartoon character that was causing the thesis program to be classified as edutainment software, as well as the pop-up dialog boxes that provided the students the area to enter their corrections. In order to streamline the students' use of the program, I replaced the pop-up dialog box with a dedicated work area that was located below the paragraph text.

After figuring out the physical location of the elements on the screen, I tackled the issue of color usage in the program. From the survey results, the students indicated that they wanted strong, clear colors with crisp text. I chose the background areas, the passive areas of the program, to be displayed in grayscale, so that it would not distract the students from their primary task of identifying and correcting errors in the paragraph. To help the students focus, I made the background areas of the paragraph and work area white to simulate the color of a sheet of paper. I also added a red boundary box around the work area to help bring and keep students' focus on the dedicated workspace. Please refer to Appendix D for a color example of the screen.

After getting the colors and location of elements of the program in place, I started revising the appearance of the user controls grouping bar. I had to turn it into something more interesting than just a black bar. It had to be aesthetic and functional. The control bar would have the hefty task of providing navigation through the program as well as giving students control of the difficulty level of the program, and giving the students a section title. Given the level of functions that would be assigned to the bar, I decided to keep the black bar, but add more organic elements to it to make it appear friendlier. I added section headers that doubled as section selection buttons in the main screen and added a white area in the bottom half of the bar for the user navigation/difficulty selection. The organic shape and black and white contrast in the design was easy on the eye and provided a clear distinction between the elements located on the bar. There were no sharp corners on the bar because I wanted to avoid anything that would distract the viewer and cause them to feel uncomfortable.

After comparing the changes to the appearance of the program, from the appearance of the paper walkthrough to the final design, I was satisfied that I had eliminated the edutainment approach by eliminating the cartoon character and the correct answer and wrong answer animations as well as simplifying the program enough that it would be successful. I also simplified the program map so that there were less complex sections and eliminated some unnecessary screens.

Next, I worked on the design problem of how the students would identify and correct the errors in a paragraph. There were several possible approaches to this. I could have the students highlight the errors in the paragraph and have the students retype the errors in the paragraph. I could have the students click on the word, and have a question appear below the paragraph asking what the correction to the error was. I could also have text entry areas where the students could retype the whole paragraph, correcting the errors as they typed. From these options, I decided that I would make the individual words of the paragraph into buttons. Each button would be assigned a value of "correct" or "incorrect." If the words were correct, it would not change, and a message would appear saying that the word was correct. If the word was incorrect, it would cause a message to appear saying that the word was incorrect and ask the student to type in the correction to the word. If the student got the word correct, the program would replace the incorrect word with the correct word colored in green to show the difference. I decided that I didn't want to have students clicking around at random, and jumping from one question if they could not answer it on the first try, so I designed the program to prevent the student from continuing until they entered the correct answer. This would require them to ask for help, or find the answer in a dictionary, thus encouraging students to interact with the teacher and other students. Please see Appendix D for examples.

After completing the layout and positioning of elements on the screen, I created the graphics in Flash and laid the graphics on their appropriate screens. After I got the graphics completed, I was ready to start programming. I knew enough programming that I could easily create the commands needed to make the buttons interactive and link to different screens, but when it came time to create the code for the interactive error selection and corrections, I found myself lacking the programming know-how to pull that off.

I asked a few friends who were programmers for help how to create the correct code for the interactive parts. They explained to me the structure of codes and showed me the correct use of variables to store user input, and how to compare the user input to the correct answer. Flash made the creation of code easy by giving us pre-set coding, but my friends were invaluable in helping me understand the correct way to program the actual interactions. After an intensive weekend of being taught, yelled at, explained to, and shown how to write correct code, I felt confident I knew enough programming to achieve my goal. I started creating the code needed for Flash 4 to become fully interactive. After working for three weeks on the programming part of the thesis project, I managed to write the code necessary to make the program work. After another two weeks of troubleshooting the program, I was satisfied that the program was finished!

Chapter 4

RESULTS

After finishing the program, I sent the thesis project to my partner for testing and reviewing. I also tested the program on some of my friends and asked people I knew to test it out on their kids.

I got very positive responses from many of the students and their parents. They appreciated the fact that the program did not talk down to the students. I designed the program to treat the students as equals. Please see appendix E for a sample e-mail response from one of the parents.

The feedback I received was positive and I also received some inquiries from parents on whether I would be interested in continuing to develop this into a full-featured software product that could be used in classroom instruction. I plan to develop this program and work with my thesis partner some more to incorporate further features into the program. The features we plan to incorporate into the "finished" version of the program are: Teacher customizable controls; Teacher defined text entry for inputting text that has customized errors to provide further tailoring to individual student needs; database of student scores; printable grade report for teacher records.

Chapter 5

SUMMARY

Self-Conclusions

After completing the thesis project, I came to the realization that my interests lie in illustration, computer graphics, and web design, rather than interactive media design and programming. This project has been incredibly fulfilling, challenging, and frustrating for me. I have learned many important lessons in the process of completing the thesis project. I have learned that I should ask people for help when I need help and suppress the urge to try to do everything by myself. I also learned that I should not make decisions without thinking about the consequences of the decisions, especially when those decisions affect my progress in school. I should have thought things through and analyzed the advantages and the disadvantages of accepting a job halfway through graduate school. The length of time required to finish this thesis is a testament to the lessons I have learned.

Program conclusions

Designing educational software has been difficult because of changing trends and themes of education, as well as shifting educational biases, as well as the ever-changing program and computer requirements. This does not include the changing stylistic fads that students go through every year. The program has received many compliments and "hands-up" sentiments from parents, teachers, and students because of its' simple, no-frills approach to the subject. Interestingly enough, some students who have ADHD said that they had an easier time focusing on this program because of the use of colors. They felt that the red boundary box around the work area helped them focus on what they were supposed to be working on in there. The limited color palette helped maintain the students' focus in the appropriate places on the screen. This proved that my program was successful in keeping students focused on identifying and correcting English errors in the paragraphs. While developing this project, I have realized that when working on interactive designs for computer software, that the concept of "simpler is better" is oftentimes the best approach. The less gadgets and fluff that one has in the program, the better chance there will be less distractions for the students, and more successful results with the students.

Chapter 6

RECOMMENDATIONS

After finishing the thesis, I took some time to step back and reflect on the completed project. After reviewing the thesis project, I have noticed some limitations that I plan to overcome as part of my continuing efforts to develop this project from a prototype to a full-fledged product. It is my hope that the program will be able to be used and updated by the teacher for continued, flexible classroom instruction. This program has the potential to assist and accentuate classroom instruction of English in the junior high and middle school levels, especially while avoiding the edutainment approach.

Some of the limitations that I have identified in the program are: the inability of the teacher to customize text paragraphs, the program's lack of statistical analysis of students' progress on the programs, and the capability to create an individualized record-keeping system that the teacher can open and print for further planning of students' individual instruction in English.

I will continue to use Macromedia Flash to develop this project. Flash has powerful ActionScripting capabilities coupled with JavaScripting commands that provide powerful controls within the program. Flash also provides cross-platform, dual web and desktop authoring capabilities and eliminates the need for individualized versions of the program. I can author once, and play many times over different computer platforms with one Flash projector file

BIBLIOGRAPHY

Education Week on the Web. 1999. *http://www.edweek.org/context/glossary/edutain.html*

Emberton, David J. and Hamlin, J. Scott. 1999. *Flash 4 Magic*. Indiana: New Riders Publications.

Heller, Steven and Pomeroy, Karen. 1997. *Design Literacy: Understanding Graphic Design*. New York: Allworth Press.

Mohler, James L. 2000. *Graphics, Animation & Interactivity with Flash 4.0*. New York: Delmar.

Reinhardt, Robert and Jon Warren Lentz. 2000. *Flash 4 Bible*. New York: IDG Books Worldwide.

Smith, Frank. 1986. *Insult to Intelligence: the bureaucratic invasion of our classrooms*. New York: Arbor House.

Smith, Frank. 1988. *Joining the literacy club*. New Hampshire: Heinemann.

GLOSSARY

Edutainment. A general classification for software programs that combine elements of instruction and entertainment, including video, animation, and music. Educators disagree on the educational value of most edutainment software (Edweek, online).

Macromedia Flash (or Flash). A vector based graphics program designed to create fast-loading web graphics and animation. It is used either with Macromedia Director or by itself to create interactive programs. Flash 4 supports limited programming with ActionScripting and JavaScripting.

Vector Graphics. Graphics that are created using mathematical formulas. Graphics created by Flash, Adobe Illustrator, and Macromedia Freehand are considered to be vector graphics. The graphics have a sharp, bright appearance and can be scaled to any size without degradation of image quality. File sizes for these kinds of graphics are very small, most times smaller than bitmap graphics.

Bitmap Graphics. Graphics that are created by arrangements of pixels on the screen. Graphics created by Adobe Photoshop, Fractal Design Painter, and Adobe ImageReady are considered to be bitmap graphics. Bitmap graphics are used for photographic images and continuous-tone images. File sizes for bitmap graphics are large and are often many times larger than vector graphics.

Projector/Player. A program that is not dependent on the presence of the authoring software to play the file. It is a self-running program that is independently playable on computers.

APPENDICES

Appendix A

Survey Questions and Results

S = U = R = V = E = Y

1. What are your favorite cartoons or comic books?

2. In these cartoons, what are the things that appeal to you the most?

3. What computer/video games do you like to play?

4. What things do you consider cool?

5. What things do you consider uncool?

6. What would make a computer/video educational game fun and exciting for you?

7. We are making a CD-ROM game for Deaf students your age that focuses on English. What areas would help you the most?

___ Spelling ___ Grammar

___ Formatting ___ Editing/Proofreading

___ Creativity/Style ___ Other

8. Do you have any ideas or advice for us? (Use the back if necessary.)

From: "some may say that I'm a dreamer..." <dab72@columbia.edu>
Subject: Survey results (for 9 so far...)
To: kws7748@ritvax.isc.rit.edu
Reply-to: dab72@columbia.edu

Hi there!!!!

Hows it going? Not bad here. Pretty busy. I start my new student-teaching assignment next Wednesday. The past 4 days I've been subbing at the middle school at Lexington, which was perfect since I could do some surveys.

So far I have 9 responses. I can easily get more. So here are the results for the first nine.

1) Favorite comic books/cartoons:

Spawn
Pitt
Taz
Garfield (3)
Archie and Jughead
Mortal Kombat (3)
Superman (3)
X-Men (4)
Batman (2)
Wolverine
Street Fighter
Predator
Spider Man

2) Reasons for liking the above:

I like the way he (Spawn) looks and the way he fights.
Pitt -- becausse beautiful drawing picture
I like the way they act
Clear colors and pictures

Mortal Kombat has secret weapons and beautiful colors and killers
Superman is strong and powerful and Batman is beautifully colored
XMen has good mutant
Comic books have many fights, tournaments, games.
Garfield has really funny stories.
...rip blood a lot and some of these are really funny and they make me laugh and cry!
Little funny and they kill people

(violent kids, aren't they?!?!)

3) Computer/video games they like:

Tekken 2 (2)
Mortal Kombat Trilogy (3)
Street Fighter Alpha 2 (3)
NBA Live 97 and 98, 99, 2000
Ultimate Mortal Kombat 3 (2)
Warcraft
Flying Tiger
Royal Rumble
X-Men Marvel
Tetris
Street Figher
Solitaire
Chips
Mortal Kombat 1,2,3
Street Fighter 1,2,3
Prime Time
Monday Night Football
NFL Live '99
NBA Game
Killer Instinct 2 Gold
Fudy Fight II
Megan Man II

Gap
Old fashioned clothes
Tetris
Soccer Champion
Drugs

6. What would make an educational game fun/exciting:

If you answer a question correctly, you can fight.
When you shoot people, have blood.
When you hit another person's head, have blood come out. Then have the sign for blood.
Vampise/Street Fight II/Vroom
Why not put mystery and if get question right, play.
Have blood.
Weapons.
Have video game with sexy women.
If you win, have a sexy woman.
Scalp.
Monsters.
Fight.
Power Magic.
Fight.

(what a morbid class..... I'm curious if the other class will respond the same way...)

7. Areas that they want to get help on:

Spelling (6)
Formatting (0)
Creativity/Style (9)
..very interesting...but HOW?
Grammar (6)
Editing/Proofreading (7)

8. Ideas, advice??

Why are you asking me?
Make the game FAST
Clear colors.
Fast game
3-D
Bright colors
Please use ASL.
Fast games.
3-D
Slow motion when you end.
Bright colors.
Please use ASL
Fast game
3-D
Slow motion when end (i have a feeling the last two kids copied each other)
I like video games that move fast and not boringly slow
Please I need in video ASL Deaf in game!!
3-d Powerful look BIG video game.
Nothing.

Okay, Kurt...here ya go!!!

Rest.

dave

Appendix B

Examples of student preferred designs

Tommy Hilfiger

Sample graphics from web site.
These samples reflect students' fascination with contrasting colors and crisp type and graphics.

Nike

Sample graphics from web site.
These samples reflect students' fascination with contrasting colors and crisp type and graphics.

Students indicated interest in the Nike Swoosh logo WITH-OUT words as being effective in getting their attention.

Polo

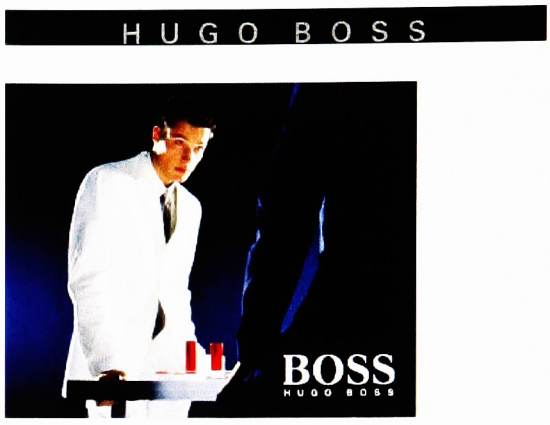
Sample graphics from web site.
These samples reflect students' fascination with status symbols. Polo is considered to be "cool" because it is one of the most expensive and stylish clothing lines available.

This choice reflects on the students' self-image and how status affects their social standing.

Boss

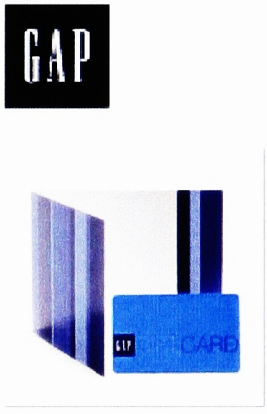
Sample graphics from web site.
These samples reflect students' fascination with status symbols. Boss is considered to be "cool" because it is one of the most expensive and stylish clothing lines available.

This choice reflects on the students' self-image and how status affects their social standing.



Gap

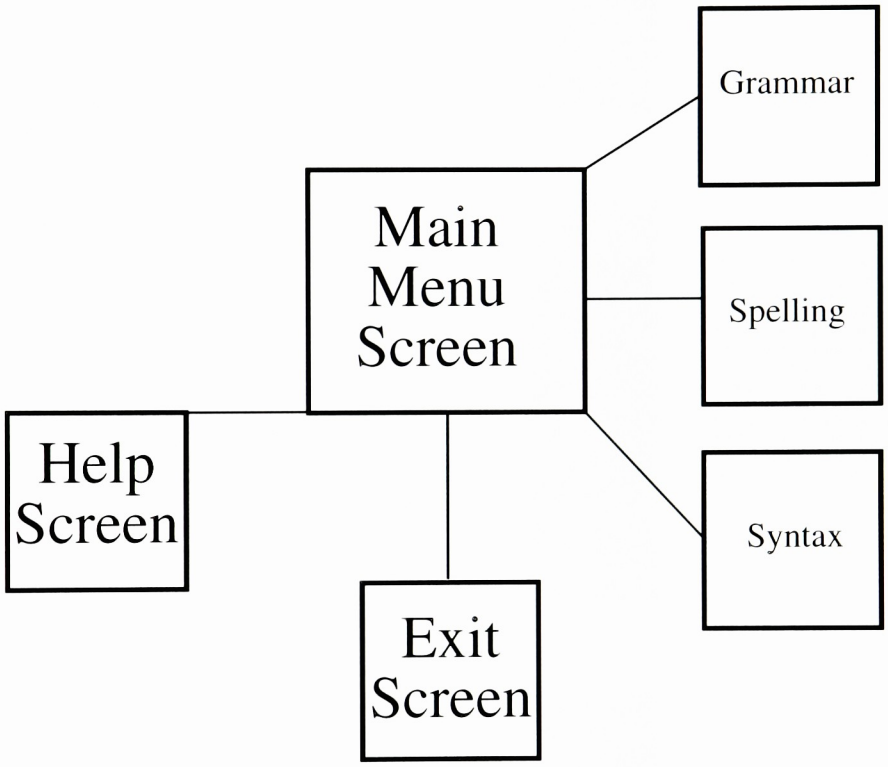
Sample graphics from web site.
These samples reflect students' fascination with status symbols. Gap is considered to be "cool" because it represents the newest trends in clothing. The colors they use are very strong and ver complementary. This helps grab students' interest and the type used shows a combination of old and new.



Appendix C

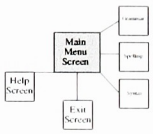
Preliminary program design

Program Map


Grammar
Main Menu Screen
Spelling
Help Screen
Syntax
Exit Screen

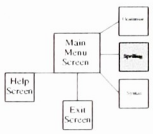
Main Menu

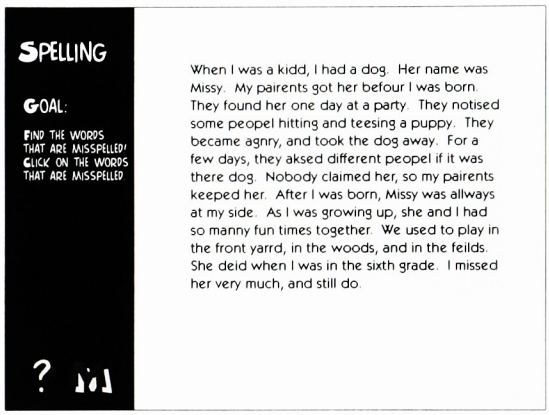
Main screen of the program. The users can navigate to any section of the program from this screen. I used the cartoon character to direct the user's attention to the navigation bar on the left of the screen.



Spelling work screen

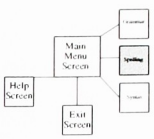
The user will identify and select the spelling errors in the paragraph by clicking on the misspelled word on the screen.

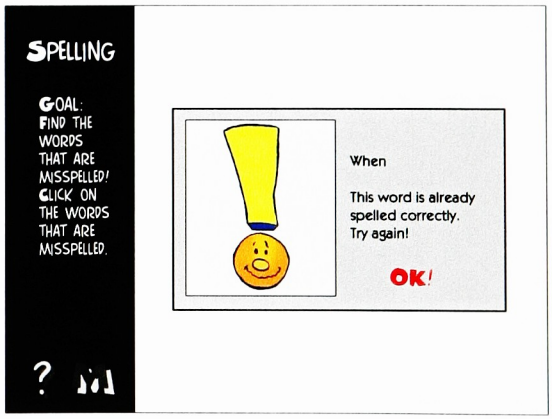




Spelling dialog box (correct word)

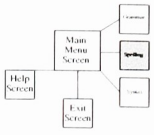
If the user selects a word that is already spelled correctly, the program will display a dialog pop-up window informing the user that the word is already correct.

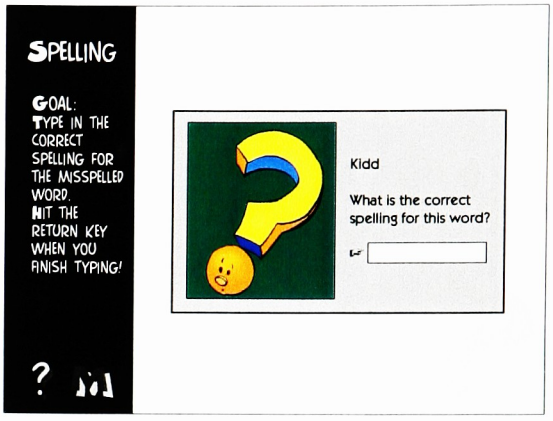




Spelling dialog box (incorrect word)

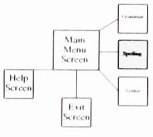
If the user selects a word that is misspelled, the program will display a pop-up dialog box asking the user to type in the correct spelling of the word. The user will be required to type their answer in the text box that is provided in the dialog box.

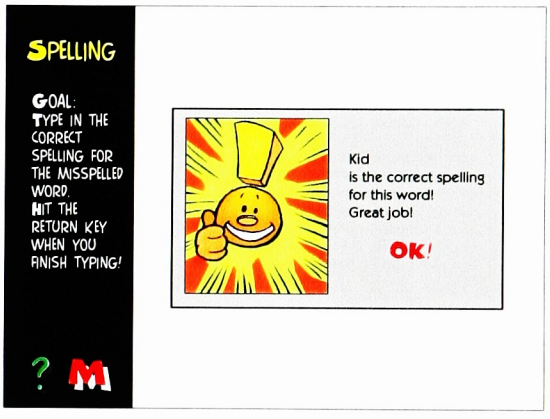




Spelling dialog box (correct answer)

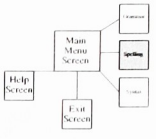
If the user enters the correct spelling, the program will display a pop-up dialog box informing the user that the answer is correct. The image in the dialog box has an animated hand that moves up and down to signify "good job!"

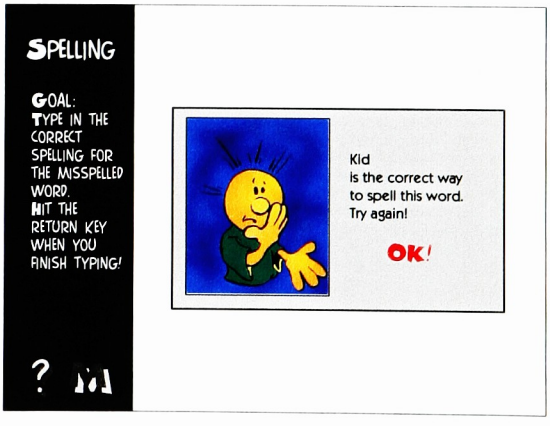




Spelling dialog box (incorrect answer)

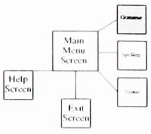
If the user enters an incorrect answer, the program will display a pop-up dialog box informing the user that the answer is incorrect. The dialog box will ask the user to try again. The program will not allow the user to continue unless they enter the correct answer.

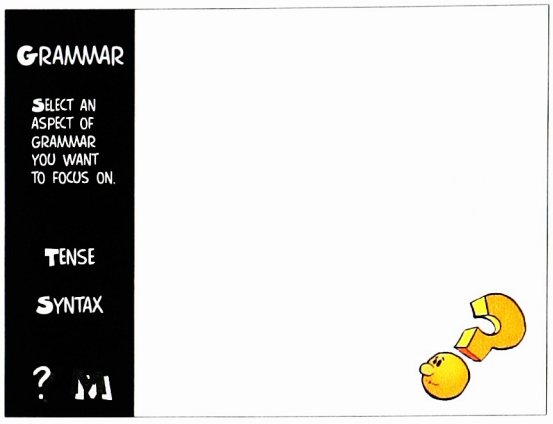




Grammar Menu

The Grammar section has two sub-sections, Tense and Syntax. This is the screen where the user can select which sub-section they want to work on.

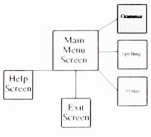


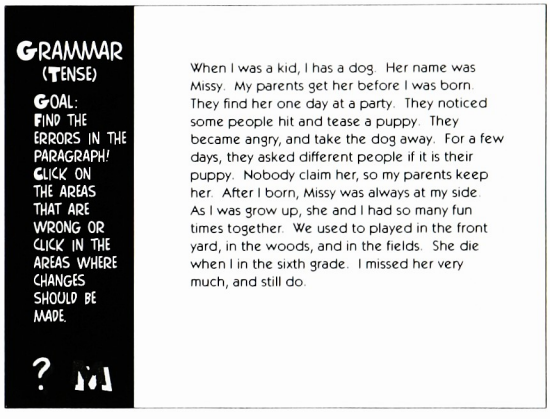


Grammar (Tense)

This is the Tense sub-section screen of the Grammar section. The user will identify and select the words that have tense errors in the paragraph.

The pop-up dialog boxes work the same way as the spelling section. The dialog boxes will require the students to type in the correct tense of the selected word.

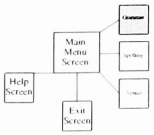


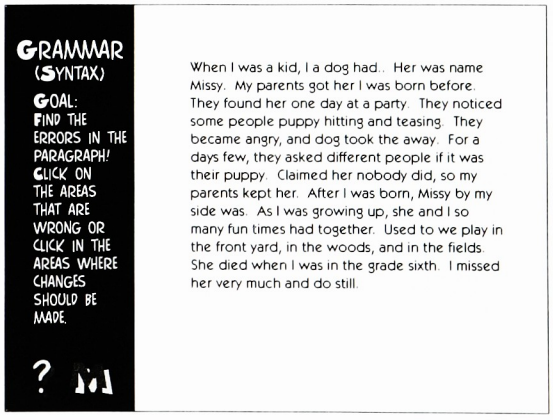


Grammar (Syntax)

This is the Syntax sub-section screen of the Grammar section. The user will identify and select the words that have syntax errors in the paragraph.

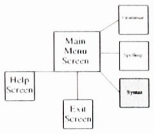
The pop-up dialog boxes work the same way as the spelling section. The dialog boxes will require the students to type in the correct tense of the selected word.

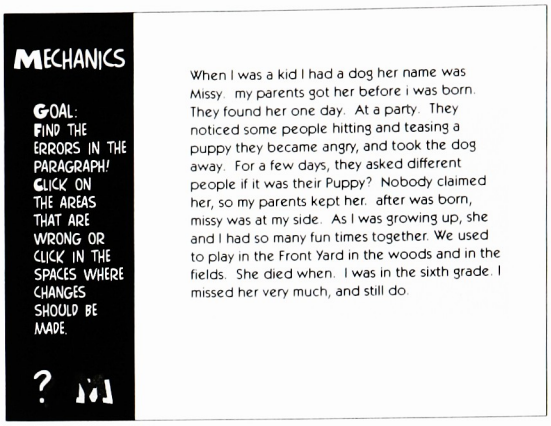




Mechanics

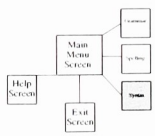
This is the mechanics paragraph screen. The user will have to identify and select the sentences that have punctuation errors in the paragraph.

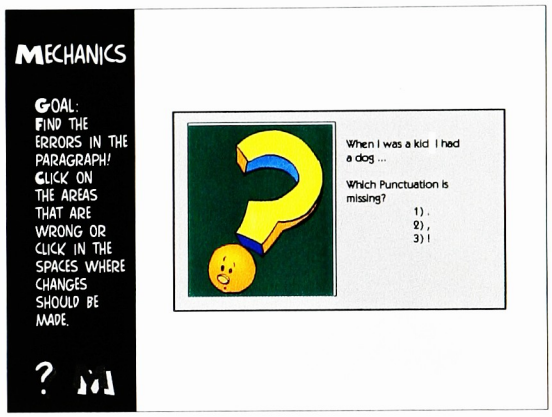




Mechanics dialog box

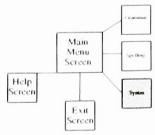
If the user selects a sentence that has a punctuation error, the program will display a pop-up dialog box. The dialog box wil ask the student to select the correct answer from a list of possible answers. For the mechanics section, the questions are all multiple choice.

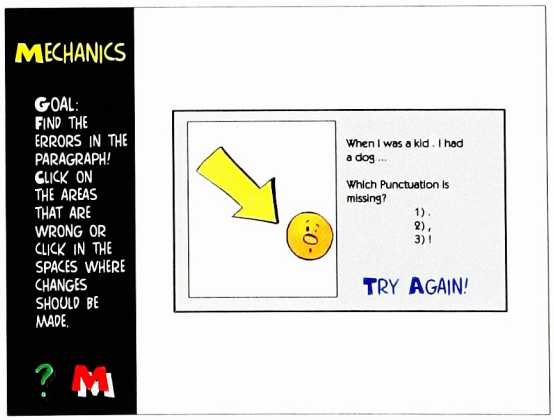




Mechanics dialog box (incorrect answer)

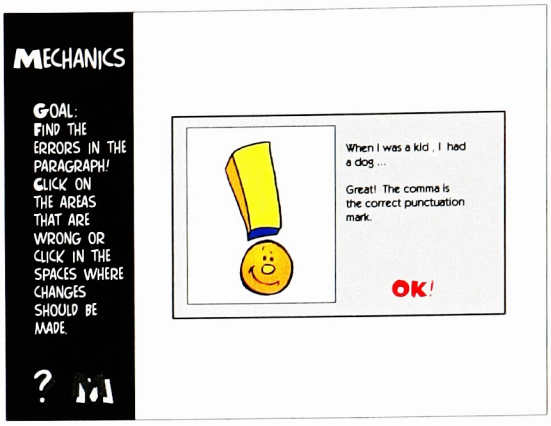
If the user selects the incorrect answer, the pop-up dialog box will inform the user that the answer is incorrect. The program will require the user to select the correct answer before allowing them to proceed.

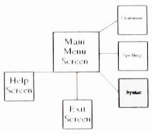




Mechanics dialog box (correct answer)

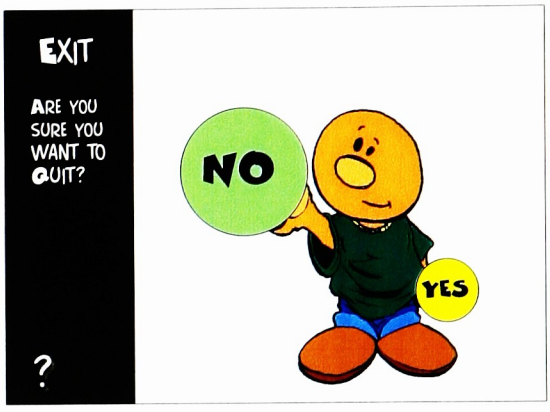
If the user selects the correct answer, the pop-up dialog box will inform the user that they got the right answer.

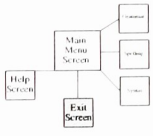




Exit Screen

If the user presses the quit button on the main screen, the program will ask the user if they are sure they want to quit. If the user wants to quit, they will need to click on the "YES" button. If they made a mistake, they can click on the "NO" button to return to the main menu screen.

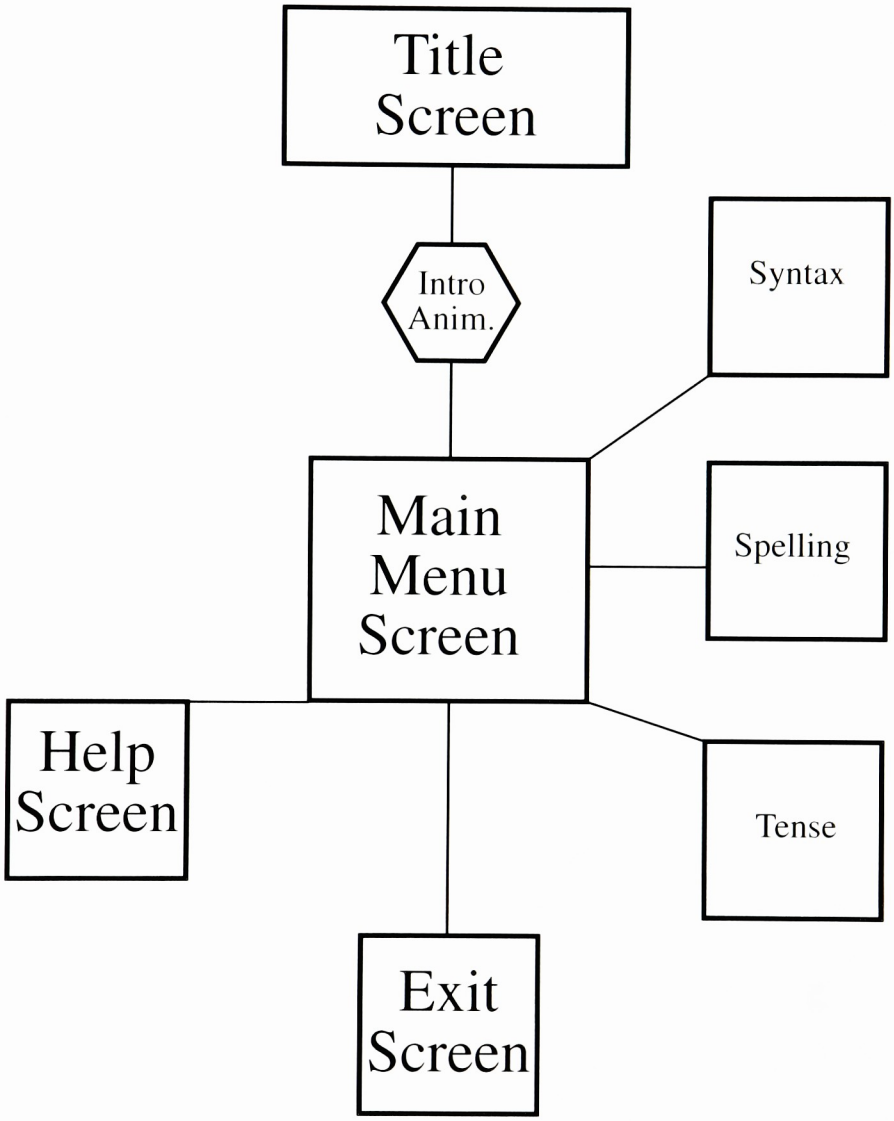




Appendix D

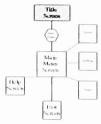
Final program design

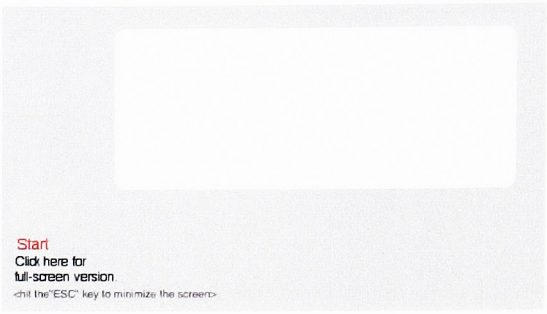
Program Map


Title Screen
Intro Anim.
Syntax
Main Menu Screen
Spelling
Help Screen
Tense
Exit Screen

Start Screen

This is the start screen. It appears when the user opens the program. This screen asks the user if they want to work in full screen mode before starting the program. Once the user is ready to start working, they can click on the "Start" button to begin.

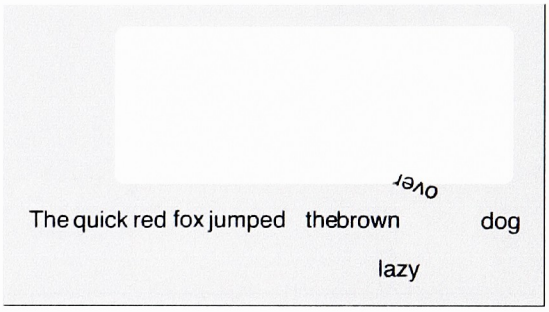




Introduction screen

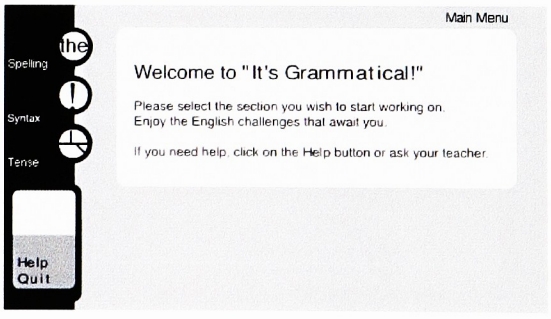
This is the introductory screen. It has an animation of a poorly written sentence being rearranged into a grammatically correct sentence.

Once the animation is complete, the program will automatically move to the main menu.



Main Menu

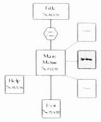
This is the main menu. This is where the user will begin using the program. The navigation bar on the left shows the different sections the user can work on. There is an introductory message that briefly explains how the program is used, as well as some tips for beginning the program.

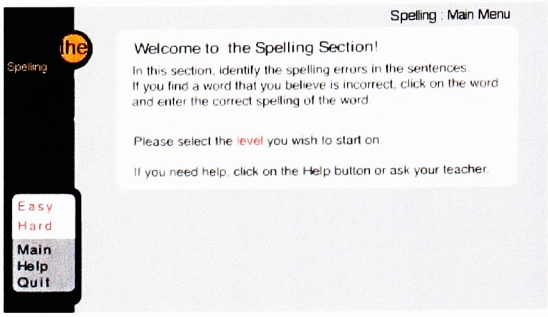


Spelling menu

This is the spelling menu screen. There are two levels of difficulty available for the spelling section. The user can choose between the "Easy" or "Hard" levels.

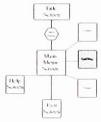
There is a brief introductory statement that has instructions for the section, as well as instructions on how to get started with the spelling exercise.

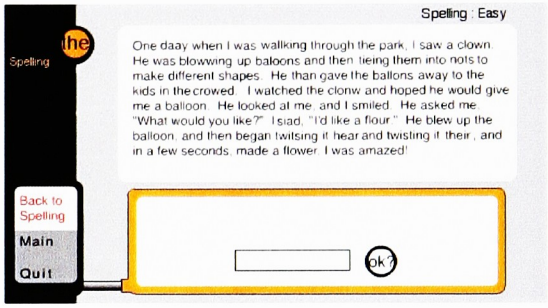




Spelling work screen (Easy)

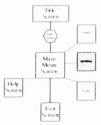
This is the spelling work screen for the "Easy" level. The user will read the paragraph and identify the spelling errors in the paragraph. Once the user has identified the spelling error, they will click on the word to enable them to answer the questions at the bottom of the screen, inside the red work area.

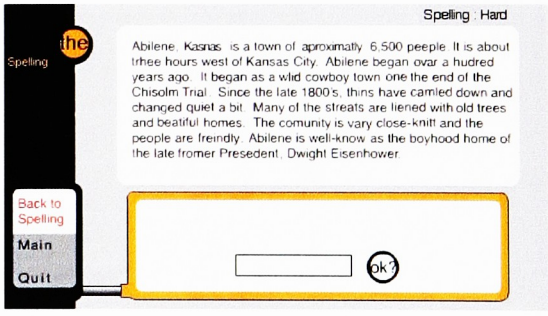




Spelling work screen (Hard)

This is the spelling work screen for the "Hard" level. The user will read the paragraph and identify the spelling errors in the paragraph. Once the user has identified the spelling error, they will click on the word to enable them to answer the questions at the bottom of the screen, inside the red work area.

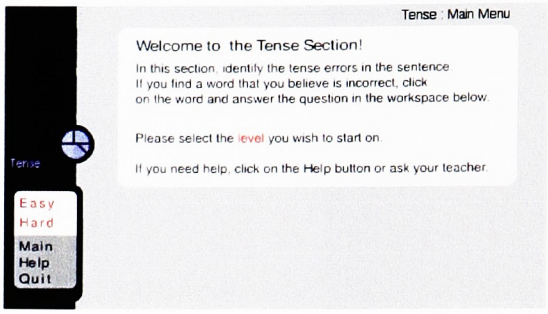




Tense menu

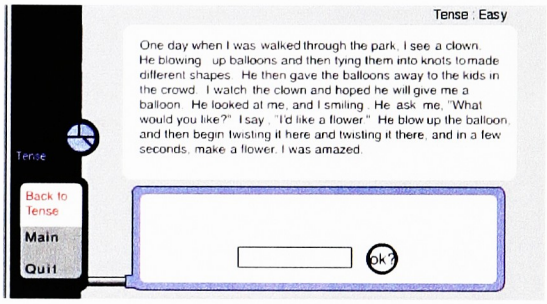
This is the tense menu screen. There are two levels of difficulty available for the tense section. The user can choose between the "Easy" or "Hard" levels.

There is a brief introductory statement that has instructions for the section, as well as instructions on how to get started with the spelling exercise.



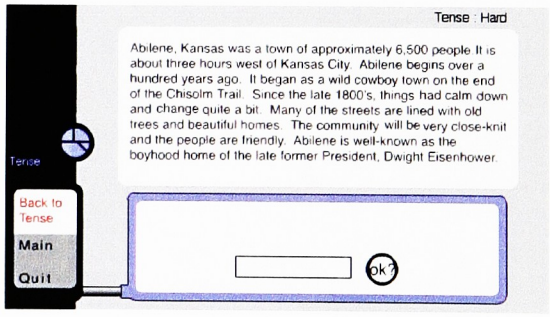
Tense work screen (Easy)

This is the tense work screen for the "Easy" level.
The user will read the paragraph and identify the tense errors in the paragraph. Once the user has identified the sentence that contains the tense error, they will need to click on the word to enable them to answer the questions at the bottom of the screen, inside the red work area.



Tense work screen (Hard)

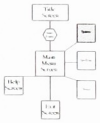
This is the tense work screen for the "Hard" level.
The user will read the paragraph and identify the tense errors in the paragraph. Once the user has identified the sentence that contains the tense error, they will need to click on the word to enable them to answer the questions at the bottom of the screen, inside the red work area.

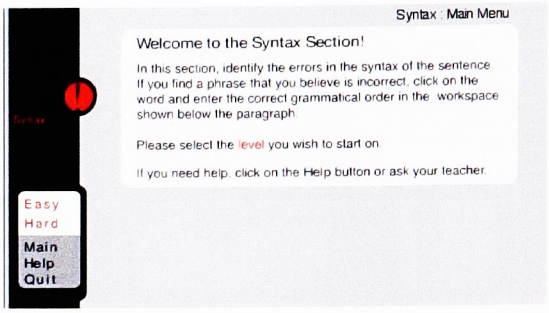


Syntax menu

This is the syntax menu screen. There are two levels of difficulty available for the spelling section. The user can choose between the "Easy" or "Hard" levels.

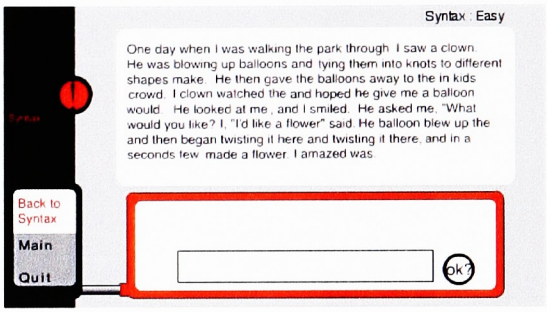
There is a brief introductory statement that has instructions for the section, as well as instructions on how to get started with the spelling exercise.





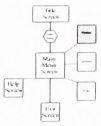
Syntax work screen (Easy)

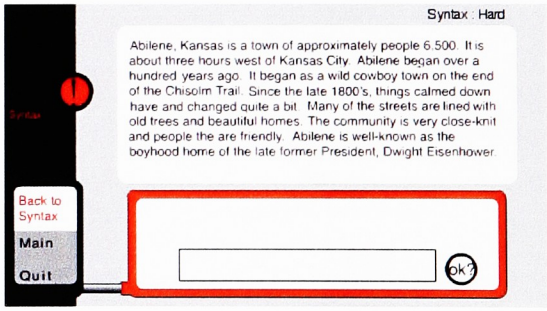
This is the tense work screen for the "Easy" level.
The user will read the paragraph and identify the syntax errors in the paragraph. Once the user has identified the sentence that contains the syntax error, they will need to click on the word to enable them to answer the questions at the bottom of the screen, inside the red work area.



Syntax work screen (Hard)

This is the tense work screen for the "Hard" level.
The user will read the paragraph and identify the syntax errors in the paragraph. Once the user has identified the sentence that contains the syntax error, they will need to click on the word to enable them to answer the questions at the bottom of the screen, inside the red work area.

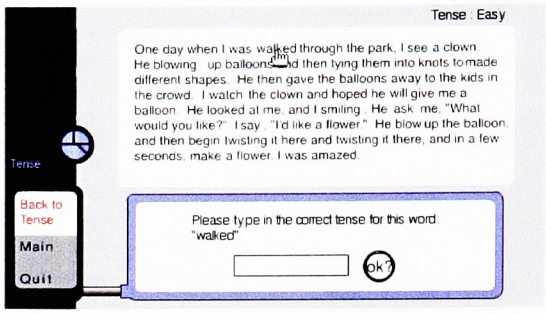




User error selection

When the user selects a word that is wrong, a dialog box will appear at the bottom of the screen stating whether the selection is correct or incorrect.

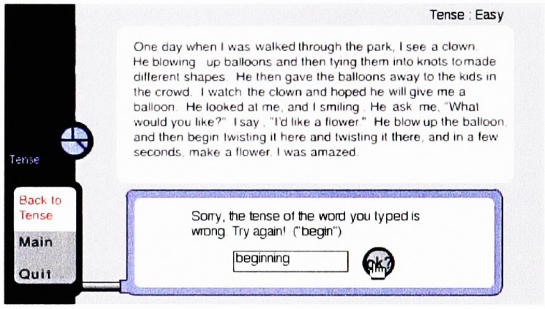
This screen shows what happens when a user selects a word that is incorrect. The dialog box asks the user to enter the correction in the text box at the bottom of the screen.



Wrong answer dialog box

If the user enters an incorrect correction and presses the "ok" button, the program will show a message stating that the correction is wrong and ask the user to try again.

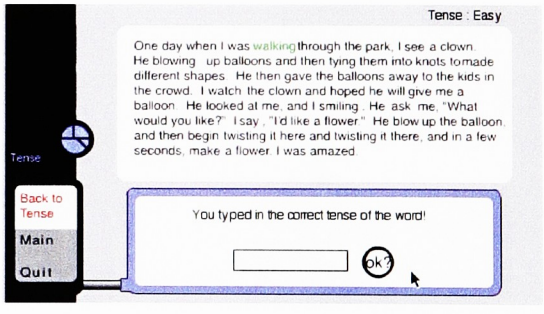
If the user does not get the correct answer, the program will not allow the user to proceed in the program until the right answer is entered.



Correct answer dialog box

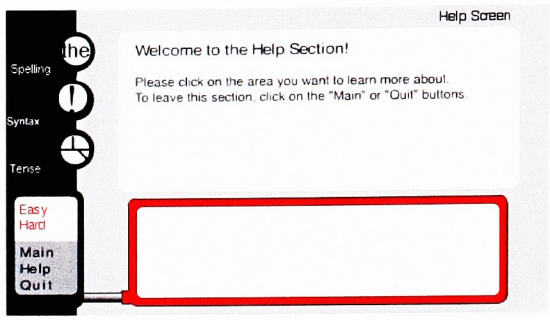
If the user enters the correct correction, then the dialog box shows a message congratulating the user. The program will also change the word that is corrected green in the paragraph.

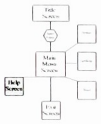
If the user selects and successfully corrects all the errors in the paragraph, the program will display a message saying, "Congratulations! You have found and corrected all errors in the paragraph." The user will then be able to proceed to the other sections of the program.



Help Screen

This is the help screen. If the user is unsure of what a button does, or what a certain part of the screen is, the user can visit the help section and click on the part of the screen they want to learn more about.

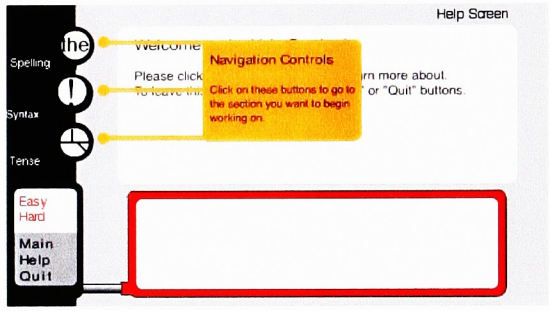


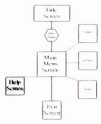


Help Screen

This is the help screen. If the user is unsure of what a button does, or what a certain part of the screen is, the user can visit the help section and click on the part of the screen they want to learn more about.

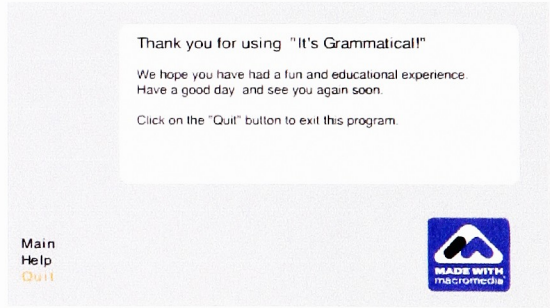
When the user clicks on part of the screen, it will trigger an animated dialog box that identifies and explains the function of the selected part. This screen shows the dialog box for the navigation area.





Exit screen.

When the user presses the quit button on the lower left corner of the screen, the program will display the exit screen. The exit screen asks the user if they want to quit. If they want to quit, they will have to press the "Quit" button again to exit the program. If the user made a mistake, they can navigate back to the program by selecting the"Main" or "Help" buttons.



Appendix E

Sample of parent feedback

Hi Kurt -

Well, I finally got my act together to show Andrew your thesis work - at 9:30 last night...it gets crazy at home, because he always has so much homework, and I get involved in that and other stuff, and just forget everything else...but anyway...here are his comments -

- the starting graphics: the moving letters he liked a lot. He tried reading them, said they went too fast, but when he realized he didn't need to read them, he thought it was cool.

- overall look of the program - he liked the fact that it was calm, not like Reader Rabbit; the design helped him to concentrate on what he was supposed to do - it was clear...

- his first choice was "spelling" - "easy level" - didn't know what the word syntax meant, but when he got into it later, understood what to do.

Now...I might note that Andrew tends to jump in to things and doesn't spend a lot of time reading directions - he is a global thinker big time...so these next comments should be tempered with that understanding...

- in the "spelling" section/easy level, he started reading the whole paragraph first, then he went back to the first misspelled word, "daay", and then WITHOUT pointing to the word, tried typing in the correct spelling - it didn't work, so he figured out he should select the word first and then type it. One comment on selecting the word - maybe it would help if the word, once selected, was highlighted. He liked the way once the word was corrected, it turned green.

Once he got past how to do it, he said "this is neat!" Some suggestions he had: for the easy level, you could mention the number of mistakes the student should find (but the hard level would NOT let you know the number of mistakes), and also, use individual sentences, rather than a paragraph format. My suggestion: you might check the level of spelling words to make sure they're consistent..."day" would be probably first grade, whereas "knot" and "tying" would probably be more third grade level...Also - (Mike's and Andrew's idea) when you have tried unsuccessfully three times? five times? to spell a word, you will be able to see the right word...

As for the easy and hard levels, Andrew suggested there be a medium level. (I know this is a pilot - but if you plan to take it any farther, this suggestion might help.)

Okay - Syntax - easy level - he tried it; first comment was that he thought the sentence was run-on, but with a little coaching, he got the idea - BUT, he typed in "i was walking through the park" and was told it was wrong because of the "i" instead of "I", I imagine. Again, he suggested that you might use single sentences for the easy level instead of paragraphs.

Misc. things -
He loved the help screens with the orange flags appearing, and he liked the way the current level was displayed when working in "spelling", "syntax", etc. He discovered also that you can change the level of difficulty by using a period, return, comma, or arrows.

Overall, Andrew really liked the program - I think it appealed to him because it wasn't "sappy" - it was fun but serious enough - not silly like most of the programs out there...

Thanks!

Kathy

Kurt,

Adam (my son-13 years old) found this (the program) to be uninteresting!

He was able to do most of the exercises, but said it was "boring!" typical for 13-year-olds who want to be visually stimulated!

Mike K.